White Paper Report

Report ID: 98875

Application Number: HD5097910

Project Director: Lynn Rainville (lrainville@sbc.edu)

Institution: Sweet Briar College

Reporting Period: 4/1/2010-3/31/2012

Report Due: 6/30/2012

Date Submitted: 8/21/2012

White Paper Report

Level 1 Start-Up Grant - NEH Office of Digital Humanities
Report ID: 98875
Application Number: HD-50979-10
Project Title: African American Families Database
Project Director: Lynn Rainville (lrainville@sbc.edu)
Institution: Sweet Briar College
Reporting Period: April 1, 2010 - March 30, 2012
Report Due: June 30, 2012
Date Submitted: August 21, 2012

Name of Project: African American Families Database
Institution: Sweet Briar College
Project Director: Dr. Lynn Rainville, Research Professor in the Humanities, Sweet Briar College, lrainville@sbc.edu, www.lynnrainville.org
Award Amount: $24,963
Project Website: www.centralvirginiahistory.org

In April 2010, Dr. Lynn Rainville and the Central Virginia Historical Researchers (CVHR) received a Level 1 Start-Up grant ($24,963) for a project that ran through March 2012 (with a hiatus between October 2010 and February 2011 when Dr. Rainville was on maternity leave). The goal of the project was to associate the antebellum and postbellum records (and identities) of enslaved individuals and freedmen.

Genealogical databases connect generations of identities by matching criteria such as names, age, birth place, sex, death date, and kinship affiliations. In many cases, African American families are at a disadvantage when searching for pre-emancipated relatives because enslaved individuals were not listed by first or last name in any pre-1870 Federal Census, and in plantation records (such as slaves listed in wills) they are often not identified with surnames. In this project we will join disparate sets of archival data (e.g., the composition of households as recorded in census records, vital records, and property tax valuations) and develop algorithms to identify patterns that can trace an enslaved individual (e.g. "John" born in 1825, enslaved by Mr. Ben Rogers) to his/her postbellum status as a free man or woman (e.g. "John Carter" listed in the 1870 census as 45 years old and a neighbor to Mr. Rogers).

The historical record contains many mentions of unique persons, but it is often difficult to establish just which unique individual is being referred to: Names can be misspelled, parts of the information may be missing, or facts may be completely incorrect. This process of disambiguating people is exacerbated when researching formerly enslaved African Americans, due to the frequent lack of recorded surnames and their absence in early census records.

As daunting as this problem might seem, there are a number of helpful aspects to keep in mind. There is indeed only one person being referred to and that person shares a set of essentially human characteristics that can be assumed: the person's death date is later than his/her birth date, his/her parents are older than he/she is, his/her mother is female, etc.

Finally, there is a rich collection of data in the historical record to draw upon- multiple decade census records, marriage, death and birth records, personal property tax records, wills, probate and other estate records and other documents specific to the enslaved population. The African American Families Database project endeavored to convert these records to on-line tables and enable researchers and descendants to track an enslaved individual and their descendants over several generations.

An explanation of the first stage of the project is available on-line: http://centralvirginiahistory.org/.

The searchable database is available: http://centralvirginiahistory.org/database1.shtml.

The geo-spatial maps are available: http://centralvirginiahistory.org/map.shtml (with a secondary link to an enhanced, searchable map).

As a result of our experiences developing the database, we have five suggestions that pertain to a wide range of digital humanities projects.

Recommendation One: Brainstorm with programmers outside of academia

The first and broadest recommendation is to brainstorm early on in your project to decide who else might be working on something similar. The field of digital humanities is still relatively new and growing. While the number of practitioners is increasing, there are still a thousand more programmer/ database administrators/ computer science majors for every one digital humanities scholar. As such, the closest parallel to your academic project may lie in other fields. In our original proposal we planned to work with two database designers who worked in firms outside of academia. Due to unrelated reasons, neither spent much time on this project but we did locate other helpful contacts in the course of developing the database.

In the case of the "African American Families Database" one of the best technological matches grew out of a program designed to catch identity fraud in the casinos in Las Vegas. While superficially different, both projects aim to uncover unique identities and associate multiple lines of evidence with individual people. Although my attention was initially turned to seemingly more likely matches — prosopography, archival repositories, and historic databases — Jeff Jonas' insights into false identities relates to the broader issue of "Identity Resolution" and "Relationship

Resolution." I am indebted to Dr. Bill Ferster, one of our team members (http://www.viseyes.org/bill.htm), for coming across Jonas' work and suggesting that I get in touch with him (to read more: http://jeffjonas.typepad.com).

I met with Jonas two times during the grant period and have plans to meet with him in the future. Our collaboration was invaluable and, amazingly, free because it enabled Jonas to test a product that he created in a novel way: locating enslaved individuals who were only listed by their given names in most antebellum records. Not only did Jonas introduce me to a huge area of research under a different name but with similar goals, he also invited me to join an exclusive group to beta test his latest product, a collaborative venture with IBM that combines the analytics power of SPSS (recently purchased by IBM) and Jonas' "modeler" technology. Between February and April 2012, I spent dozens of hours learning the program and testing it with my 19th-century data. Although the program was designed for 21st categories (such as social security numbers and addresses), the algorithms behind the data categories encouraged me to think of new ways to correlate identities from historic records.

During the period of the start-up grant, Jonas agreed to work with me to test IBM SPSS Modeler and the Entity Analytics plugin, which adds social network analysis capabilities:

http://www-01.ibm.com/software/analytics/spss/products/modeler/

We somewhat unwittingly stumbled across an unusual application for it while normalizing data from the 1870 census: we used it to look for duplicate records. While less common in today's census, in the 19th century census takers were on horseback, riding through rural regions with few detailed and accurate maps. Given the scale of just one rural, Virginian county (Albemarle in 1860) with over 26,000 people, it is not surprising that mistakes were made, but locating them 150+ years later is a different matter.

We were excited to get this initial proof that his program could resolve ambiguous (or in this case duplicate) identities in historic records.

Recommendation Two: Make certain your programmers are familiar with Open Source software and with documenting their work

For most applicants, this may be obvious, but if you have a Phase I start-up you may not be fully aware of the differences among open source, free ware, or share ware. Just because the word "free" or "share" is in the title, doesn't mean that the code is readily accessible, shareable, or even free. Lots of "free" templates require fee-based add ons to access all of the program's features or the template is free but in order to make changes to the code you need to pay the author a fee.

For example, we planned to use a SQL-based database to organize the archival information about African American families. We hired someone to develop an open-source, MySQL database. But the corollary requirement was to ensure that the developer was willing to have his code shared. In other words, a company can use an open-source product to develop a proprietary database. For our project it was important that we could share the final template with other counties so that they could upload their records into a larger database.

The second thing to ask your designer is whether he/she has experience creating open source code. Ideally the product that you create will be registered with one of the well-known open source "hubs," such as Github (https://github.com/) or SourceForge (http://sourceforge.net/). By uploading your code to these sites, other programmers can make recommendations and edit your code, resulting in collaborations among a wider group of programmers. It also serves as a repository for storing your code and recording any changes in it as you develop your final product.

Finally, you should include money in your budget for the programmer to document his/her code and edits. A lot of programmers are happy to spend the time developing new code, but less willing to document what they have done. Both steps are important in creating a foundation that can be edited by others in the future.

Recommendation Three: Use high-tech solutions as necessary, but first normalize your data

During my first meeting with Jeff Jonas he provided me with low tech, but exceedingly useful suggestions for how to organize naming data into spreadsheets. This is prior to analyzing the relationships among the data; but this data normalization helps lay a strong foundation for the application of algorithms or any predictive coding. For example, he suggested converting all first names into their complete (non-nickname) form, “Nick” becomes “Nicholas,” and “Bettie” becomes “Elizabeth.” While time consuming, this means that the instability of first names over a lifetime does not hinder the ability to locate them in searches. Perhaps somewhat counterintuitive, the next step is to shorten the names by removing duplicate consonants (in a row, e.g., “Matthew” becomes “Mathew”). This decreases the chances of missing someone because of a mis-spelled name or an abbreviation. In Jonas’ case, he has spent years and lots of money developing very complicated identity resolution databases that take this even further and remove vowels (a common source of mistaken spelling) and can handle multi-national names (a useful feature when studying the given names of 18th-century enslaved individuals). In the

end, in this project we experimented with the elongated names but ran into difficulties deciding the 'origin' name for some African-influenced slave names, such as "Mourning," or "Nicey."

Note: the complication with locating antebellum, enslaved individuals is that surnames were not regularly recorded in written records, so often the challenge is trying to decide whether a "Robert" on one plantation in 1850 is the same "Robert" (or "Bob" or "Robbie") on another plantation in 1860.

Jonas had a second strategy for locating two individuals from different datasets, creating regular strings from common types of data. For example, you might create a family string, for Robbie and his wife Mary from an 1850 record: "MaryRobert" (another suggestion was to alphabetize the names in any list so that they always occurred in the same order even if the list wasn't complete). Then, if you find a record from 1860 which lists a married couple, Robert and Mary, and their two children, you add on to the string: "MaryNiceyPrestonRobert." Finally, if you locate a post-bellum record, say the 1870 Census, with a household that contains a widower and his children, you would have "NiceyPrestonRobert" (in this case you would list the surnames in a different column). If you build a spreadsheet of first names using this approach you can do simple sorts to narrow down identical individuals from different datasets. No fancy algorithms or complicated programming is needed, just the patience to create the strings. This was one powerful way to sort the thousands of individuals that we used in our sample dataset for the start-up grant.

Recommendation Four: Consider developing add-ons for existing technologies instead of creating new products

In some cases you can combine existing technologies instead of creating a brand new product. For example, we used Google Maps, ArcGIS, and Mappetizer to map our geo-spatial data. Our goal was to use post-emancipation records (such as an 1867 list of white landowners who had recently freed African Americans living on their land) to help us associate postbellum African Americans with their antebellum plantations. Immediately after the Civil War, many African Americans continued to live on plantations where they were previously enslaved. We layered this geo-rectified data with more recent maps from the segregation years of 1941 and 1942, which still listed land ownership by race. Although separated by almost a century, some black families continued living on the same lands generation after generation. Most importantly, the 20th century data illustrated African American neighborhoods, many of which were based on antebellum concentrations of plantations.

To map this data we tried two approaches, from the simpler option (google maps) to a more complex one (ArcGIS). The quickest, cheapest and simplest option was a google map which used personal property tax records from 1867, 1868, and 1869 to create polygons for postbellum white landowners and then we inserted "pins" for black taxpayers (whose locations were recorded by referring to the nearest white landowner). This resulted in a very useful on-line map that also associated postbellum African Americans with their former owners: http://centralvirginiahistory.org/map.shtml.

The drawback to this map is that it wasn't searchable and it wasn't practical to include multiple layers. In order to fully analyze the geo-spatial data we decided to build a GIS-based map. We were very lucky to have a GIS-expert in our group of local historians (Robert Vernon, who works at the nearby National Ground Intelligence Center). He built a layered map with environmental (streams, lakes, soils, etc.), historical (schools, roads, and train tracks), and spatial data (historic maps from various decades). He built on his ArcGIS training to create the layers and then he installed Mappetizer, an ArcGIS extension, to make the map interactive without requiring expensive server software or server-side support. In the end, the map turned out to be a very complicated product both in terms of the GIS skills required, the countless person-hours required to geo-rectify old maps and add in data by hand, and the troubles involved in making the searchable map work on all browsers. The map required SVG functionality and during the course of the grant Firefox upgraded to version 13 (which doesn't support complete SVG functionality), Internet Explorer proved to be very slow, Opera worked okay but mouse clicks took a long time to produce results, and Chrome was fast but not fully functional. Finally, we turned to flat files as a compromise that provided functionality (albeit not a uniform experience across all browsers) but lacked a robust performance.

The next stage of our geo-spatial work would be finding an affordable solution to hosting large amounts of graphic data (e.g., hundreds of photographs that depict the location of black homes), making it searchable on-line, and finding a product that it fully supported across multiple browsers.

Recommendation Five: Leave money for fixing post-release bugs

As any programmer knows, whenever you release a new product you can expect bugs. In academia, the tendency is to work on a product (say a paper) up until the last minute and then 'release it.' This strategy does not work when releasing new programs. Because I had a Phase I NEH Digital Humanities start-up grant, there was not a lot of time to work through the development and release of the code for the database. I focused the time of our database designer on adding new features, not trying to 'break' existing ones. I released a beta version of the on-line database about a month before the end of the grant period (in a two-stage process, first to a limited group of researchers and then a larger one). In a related piece of advice, realize that no matter how excited you are about the release of your product, it may take your audience a while (weeks….months….even longer) to get around to using it. In our case I had released the beta version to about 100 people. Of those, about six signed up to use the password-protected site (note: requiring an email address to sign-in to a free product enables you to keep track of who is using it). As you can imagine, six testers is not enough to catch bugs. Instead, minor issues (like login problems) did not surface until months after the end of the official grant which made it difficult to find the monies to make necessary fixes. In our defense, we had released an early beta version of the database six months earlier, but the final version, as discussed above, was so radically different that it produced its own set of bugs.